|  |
| --- |
| algorithm 1：Convolution-based Multi-dimensional Conditional Position Encoding |
| Input: A bag of features embedding  (N is determined by the number of patches in a WSI and the number of key features after feature extraction, and it is dynamically changed)  class CCPE(nn.Module):  def \_\_init\_\_(self, dim=512):  super(CCPE,self).\_init\_()  self.proj = nn.Conv2d(dim, dim, 1, 1, 1//2, groups=dim)  self.proj1 = nn.Conv2d(dim, dim, 5, 1, 5//2, groups=dim)  self.proj2 = nn.Conv2d(dim, dim, 3, 1, 3//2, groups=dim) |
| def forward(self, x, H, W):  B, \_, C = x.shape  cls\_token, feat\_token = x[:, 0], x[:, 1:]  cnn\_feat = feat\_token.transpose(1, 2).view(B, C, H, W)  x = self.proj(cnn\_feat)+cnn\_feat+self.proj1(cnn\_feat)+self.proj2(cnn\_feat)  x = x.flatten(2).transpose(1, 2)  x = torch.cat((cls\_token.unsqueeze(1), x), dim=1) |
| Output: a bag of instances of vector representation after extracting position-encoding information |